PATENT ABSTRACTS OF JAPAN

(11)Publication number : 2001-175626
(43)Date of publication of application : 29.06.2001

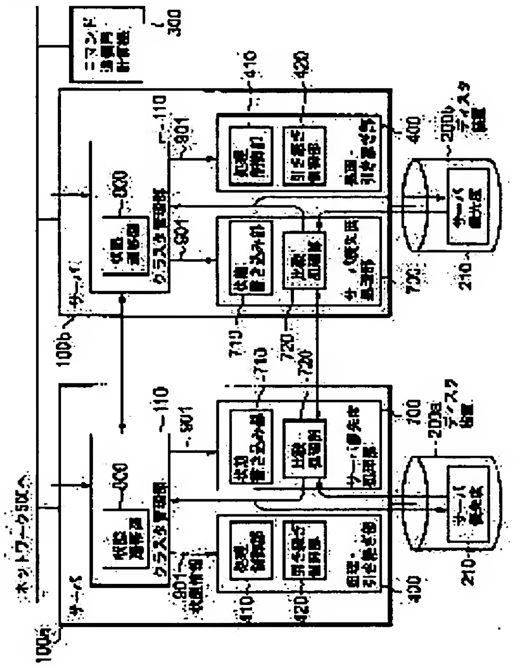
(51)Int.CI. G06F 15/177
G06F 11/20

(21)Application number : 11-364571
(22)Date of filing : 22.12.1999

(71)Applicant : TOSHIBA CORP
(72)Inventor : YAMAMOTO KOJI
ENDO KOTARO

(54) RECORDING MEDIUM WITH RECORDED PROGRAM DECIDING SERVER COMPUTER HAVING PERFORMED PROCESSING FINALLY AND HIGH- AVAILABILITY COMPUTER SYSTEM

(57)Abstract:
PROBLEM TO BE SOLVED: To actualize automatic operation by eliminating the limit of the frequency of hand-over of a processing and to improve decision precision by deciding a server having performed a process finality by using only local information.
SOLUTION: When a fault occurs in at least one server 100a or 100b and when the server recovers from the fault, the cluster management part 110 of the server shifts server state according to a state transition chart 800. A state write part 710 records server priority 210 determined by the server state that state change information 901 indicates on disk drivers 200a and 200b. A comparing processing part 720 decides whether or not its server has higher priority according to at least the priority 210 of its server if a fault occurs to the servers 100a and 100b and at least its server recovers from the fault thereafter. The cluster management part 110 makes a corresponding state shift according to the decision result and state transition chart 800.



LEGAL STATUS

[Date of request for examination]	15.09.2004
[Date of sending the examiner's decision of rejection]	
[Kind of final disposal of application other than the examiner's decision of rejection or application converted registration]	
[Date of final disposal for application]	
[Patent number]	
[Date of registration]	
[Number of appeal against examiner's decision of rejection]	

CLAIMS

[Claim(s)]
[Claim 1] When a failure occurs by the server calculating machine which is processing by one side of two server calculating machines processing It is applied to the sex computing system for Takayoshi with another server computer able to succeed processing. It is the record medium which recorded the program which judges the server calculating machine which was processing at the last for performing said each server calculating machine and in which calculating-machine reading is possible. The state-transition step which performs the state transition of the server calculating machine concerned according to a predetermined state transition diagram when it returns to said each server calculating machine from the case where a failure occurs at least in one side of said two server calculating machines, and a failure, The server priority record step recorded on the store in which a self-server calculating machine has locally the server priority decided by the condition of the self-server calculating machine after the transition concerned at least according to said state transition, When a failure occurs to said both two server calculating machines and a self-server calculating machine returns from a failure at least after that The priority judging step which judges whether the direction of the self-server calculating machine concerned has a high priority based on the server priority recorded on said store of the self-server calculating machine concerned at least, Based on the priority judging result in said priority judging step, by judging whether it is the server calculating machine by which the self-server calculating machine was processing at the end The record medium which recorded the program which judges the server computer which was processing at the last for performing the final-treatment computer judging step to which determine whether a self-server computer continues processing and the state transition in said state-transition step is made to perform by the decision result.
[Claim 2] Said priority judging step measures the server priority recorded on said store of both the servers calculating machine concerned, when a failure occurs to said both two server calculating machines and both servers calculating machines of both return from a failure after that. The record medium which recorded the program which judges the server computer which was processing at the last according to claim 1 characterized by including the 1st priority judging step which judges whether the direction of a self-server computer has a high priority.
[Claim 3] A failure generates said priority judging step to said both two server computers. The highest priority judging step which judges whether the server priority recorded on said store of a self-server calculating machine is the highest priority when only a self-server calculating machine returns from a failure after that, Only when judged with it being the highest priority at said highest priority judging step, the direction of a self-server computer the server computer which was processing at the last according to claim 2 characterized by including further the 2nd priority judging step judged as a priority being high The record medium which recorded the program to judge.
[Claim 4] Although processing is performed with the "master" in which the partner to whom said state-transition step takes over the processing which processes and is performing the condition of each of said server computer exists The "single master" in which the partner who succeeds the processing which requires a line does not exist, The "slave" which has received the information for taking over although not processed, It does not process. And it classifies into four conditions of a "halt" of having not received the information for taking over. The record medium which recorded the program which judges the server calculating machine which was processing at the last according to claim 3 characterized by performing the state transition of said two server calculating machines according to said created state transition diagram.

[Claim 5] A failure generates said state-transition step to said both two server computers. When both servers computers of both return from a failure after that, it is based on the judgment result and said state transition diagram in said final-treatment computer judging step. The 1st state-transition step which performs the state transition to which one side of said two server computers will be in said single master condition from said idle state, and another side will be from said idle state in said slave state, The record medium which recorded the program which judges the server computer which was processing at the last according to claim 4 characterized by including the 2nd state-transition step which performs the state transition from which said one side will be in master mode while the account another side of back to front has been a slave state.
[Claim 6] A failure generates said state-transition step to said both two server computers. When only a self-server computer returns from a failure after that, it is based on the judgment result and said state transition diagram in said final-treatment computer judging step. The 3rd state-transition step which performs the state transition to which a self-server computer will be in said single master condition from said idle state, or maintains a current condition, [whether when a partner server calculating machine returns from a failure after that, the state transition to which the partner server calculating machine concerned will be from said idle state in said slave state is performed, and] Or one side of said two server computers will be in said single master condition. The record medium which recorded the program which judges the server computer which was processing at the last according to claim 4 characterized by including the 4th state-transition step which performs the state transition from which another side will be in said slave state.
[Claim 7] Since the server priority recorded on said store of the self-server calculating machine concerned is not the highest priority when a failure generates said state-transition step to said both two server calculating machines and only a self-server calculating machine returns from a failure after that In the condition that cannot perform a judgment at said highest priority judging step, but the self-server computer concerned continues processing and that it cannot judge with it being a computer When the compulsive initiation instruction which makes processing continue compulsorily to the self-server computer concerned is given from the outside The record medium which recorded the program which judges the server calculating machine by which the self-server calculating machine concerned was processing at the last according to claim 3 characterized by including the compulsive initiation step to which the state transition of the self-server calculating machine concerned is made to perform as a server calculating machine which continues processing.
[Claim 8] In the sex computing system for Takayoshi with another server calculating machine able to succeed processing when a failure occurs by the server calculating machine which is processing by one side of two server calculating machines processing A state-transition means to perform the state transition of the server calculating machine concerned according to a predetermined state transition diagram when said each server calculating machine returns from the case where a failure occurs at least in one side of said two server calculating machines, and a failure, A condition write-in means to record on the storage in which a self-server calculating machine has locally the server priority decided by the condition of the self-server calculating machine after the transition concerned at least whenever a state transition is performed by said state-transition means, When a failure occurs to said both two server calculating machines and a self-server calculating machine returns from a failure at least after that A priority judging means to judge whether the direction of the self-server calculating machine concerned has a high priority based on the server priority recorded on said store of the self-server calculating machine concerned at least, Based on the priority judging result in said priority judging means, by judging whether it is the server calculating machine by which the self-server calculating machine was processing at the end The computing system characterized by providing a final-treatment computer judging means to determine whether a self-server computer continues processing and to make the decision result perform the state transition by said state-transition means.

[Translation done.]

DETAILED DESCRIPTION

[Detailed Description of the Invention]
[0001]
[Field of the Invention] Even if a failure occurs by the server calculating machine which is processing by one side of two server calculating machines processing, this invention The sex computing system for Takayoshi with which another server computer can succeed processing is started. When a failure occurs especially to both server computers and at least one side returns from a failure after that It is related with the record medium which recorded the program which judges the server calculating machine which was processing [required to judge whether the calculating machine which returned continues processing] at the last, and the sex computing system for Takayoshi.
[0002]
[Description of the Prior Art] Even if it combines two or more server calculating machines (it is hereafter called a server for short) (two sets of for example, servers) in a network etc. and a failure occurs in one server from the former, when another server succeeds the processing (service) suspended with the failure, the failure-proof computer system of the cluster type which enabled it to maintain availability as the whole system, i.e., the sex computing system for Takayoshi, is developed variously.
[0003] While performing processing by one server in the system equipped with shared memory equipments, such as shared disk equipment, among this kind of computer systems, it is common to record information required to succeed processing from the server concerned on the shared memory equipment concerned.
[0004] Even if it is which failure return server in using the information which was recorded by shared memory equipment in any [when only one when a failure occurs, for example in two sets of servers in the computer system of such a configuration and both servers of both return from a failure after that of servers returns from a failure] case, it can take over easily.
[0005] However, a system without shared memory equipment is also in the sex computing system for Takayoshi. In order to enable taking over of processing in such a computer system, while performing processing by one server, it is common to send information required to carry out a continuation of processing from the server concerned to the server of another side.
[0006] thus, when are carried out, and a failure occurs in the server which is performing processing and it becomes impossible to continue processing, it becomes possible to continue processing using the information which was alike till then and has been sent by the server of another side, i.e., to succeed processing.
[0007] However, when a failure occurs in both servers, it is not easy to take over the above-mentioned processing. The reason is [both] that it must determine whether to be that which server processes (it succeeds) if it is the case where for example, both servers carry out a failure return (restoration). Moreover, it is because it must judge whether it is processing by the server which returned (it succeeding) if it is the case where only one of servers returns from a failure. This conventional technique is explained in full detail below.
[0008] When a failure occurs in two sets of servers and both servers of both carry out a failure return, in order for the server (slave) which was not processing by then to continue processing, it is necessary to get the information for taking over from the server (master) which was processing.
[0009] However, when the information for taking over is not sent from this server to another server (slave) just before a failure occurs in the server (master) which was processing at the end, processing cannot be continued in concerned another server (slave). Therefore, when both servers of both carry out a failure return, it is necessary to determine the server of the direction which was processing at the end as a server which succeeds

processing (selection).
[0010] Next, when the information for taking over is sent from this server to another server (slave) and both servers carry out [both] a failure return before the failure occurred in the server (master) which was processing at the end, whichever it chooses a server, it is possible to carry out a continuation of processing. However, about the processing in front of failure generating, the failure may have occurred, before sending information required for taking over to one more set of a server from the server which was processing at the end. Therefore, also in here, it is necessary to choose the server of the direction which was processing at the end.
[0011] Moreover, also when only one of servers returns from a failure, both the conditions that make the server process are the server to which the server was processing to the last two sets of inside for the same reason as the case where both servers carry out a failure return.
[0012] Therefore, either of two approaches described below was applied so that the server which was processing at the end could be judged conventionally.
[0013] (1) Decide a primary server and another server to be secondary servers for eye ****** limit taking over at a time, and one one of servers, and a primary server starts a master and a secondary server surely starts actuation as a slave in the beginning. Here, a master processes (processing demanded from the client) and sends the information for taking over to a slave. A slave is saved at external storage (local external storage), such as a disk unit which receives the information for the taking over sent from a master, and self has.
[0014] And if it takes over to a primary server by a failure occurring once, even if a primary server returns after that, it will not use. That is, taking over is limited at a time. In this case, if a secondary server will record whether self processed or not on own external storage, it can judge which was processing easily till recently.
[0015] However, this conventional technique cannot realize the so-called unattended operation [say / continuing processing as much as possible], whenever failure generating or a failure return may take place to the server of one of the two or both, since taking over is once possible.
[0016] (2) Set the clock (time of day) of two sets of eye ****** use time information, and servers. When a server starts processing, it records on the external storage in which self has current time. The information on the time of day recorded on external storage is mutually delivered and received through a network, and when are done in this way and both servers of both carry out a failure return, a server with newer time information is judged as a server which was processing at the end.
[0017] The approach using this time information assumes a global thing called time amount tacitly, and assumes that the clock which each server uses always synchronizes. However, it does not restrict that the actual clock always not necessarily synchronizes, but this approach has a problem in integrity. Moreover, since time information cannot be delivered and received between another servers when only one server carries out a failure return, self cannot judge whether it is the server which was processing at the end.
[0018]
[Problem(s) to be Solved by the Invention] As described above, even if a failure occurs in the server which is processing by one side of two sets of servers processing In the conventional sex computing system for Takayoshi which is a sex computing system for Takayoshi with another server able to succeed processing, and does not have shared memory equipment When a failure occurred in two sets of servers and at least one side returned from a failure after that, the "approach of limiting taking over at a time" or the "approach of using time information" was applied as an approach whose judgment of the server which continues processing, i.e., the server which was processing at the last, is enabled.
[0019] However, by "the approach of limiting taking over at a time", since taking over was once possible, there was a problem that unattended operation was unrealizable. Moreover, although a global thing called time amount was assumed tacitly and it assumed that the clock which each server uses always synchronizes by "the approach of using time information", the actual clock did not necessarily restrict always synchronizing, therefore the problem was in judgment precision (integrity). Moreover, when only one server carried out a failure return, the problem that it could not judge whether it is the server which was processing at the end also had self.
[0020] This invention is what was made in consideration of the above-mentioned situation. The purpose While losing a limit of the count of processing taking over and realizing unattended operation by not using global information of time information, but judging the server which was processing at the end only using local information It is in offering the record medium which recorded the program which judges the server (server

computer) which can aim at improvement in judgment precision, and which was processing at the last, and the sex computing system for Takayoshi.
[0021]
[Means for Solving the Problem] When a failure occurs in the server which is processing by one side of two sets of servers processing, the record medium of this invention Are applied to the sex computing system for Takayoshi with another server able to succeed processing. It is the program which judges the server which was processing at the end. To each above-mentioned server The state-transition step which performs the state transition of the server concerned according to a predetermined state transition diagram when it returns from the case where a failure occurs at each following step, i.e., at least one side of the two above-mentioned sets of servers, and a failure, The server priority record step recorded on the store in which a self-server has locally the server priority decided by the condition of the self-server after the transition concerned at least according to this state transition, When a failure occurs in both servers and a self-server returns from a failure at least after that The priority judging step which judges whether the direction of the self-server concerned has a high priority based on the server priority recorded on the store of the self-server concerned at least, Based on the priority judging result in this priority judging step, by judging whether it is the server to which the self-server was processing at the end It determines whether a self-server continues processing and is characterized by recording the program which judges the server which was processing at the last for performing the final-treatment calculating-machine judging step to which the state transition in the above-mentioned state-transition step is made to perform by the decision result.
[0022] In such a configuration, a state variable called the server priority decided by transition in the condition of following a predetermined state transition diagram is recorded on local storage, and it becomes possible to lose a limit of the count of processing taking over and to realize unattended operation from the server which was processing to the last being judged using the state transition concerned and a local state variable.
[0023] It is good to give the 1st priority judging step which judges whether the direction of a self-server has a high priority in the server priority of both the servers concerned as compared with the time of returning to the above-mentioned priority judging step from a failure for the processing at the time of a failure occurring in the two above-mentioned sets of both servers, and both servers of both returning from a failure after that here, both the following steps, i.e., both above-mentioned servers.
[0024] Moreover, for the processing at the time of a failure occurring in the two above-mentioned sets of both servers, and only a self-server returning from a failure after that The highest priority judging step which judges whether the following two steps, i.e., the server priority of a self-server, are the highest priorities to the above-mentioned priority judging step, Only when judged with it being the highest priority here, it is good to give the 2nd priority judging step judged as the direction of a self-server having a high priority.
[0025] Moreover, although processing is performed with the "master" in which the partner who succeeds the processing which this invention processes as a condition of the two above-mentioned sets of servers shown with the above-mentioned state transition diagram, and is performed exists The "single master" in which the partner who succeeds the processing which requires a line does not exist, Although not processed, it is characterized by applying the combination in four conditions of a "halt" of not processing with the "slave" which has received the information for taking over, and having not received the information for taking over.
[0026] By thus, the thing which it classifies into the "master" in which the partner who succeeds the processing which is performing the server which processes exists, and the "single master" in which the partner who succeeds the processing currently performed does not exist, and is distinguished It prevents that the local state variable (server priority) of both servers serves as the same value, and it becomes possible to improve the judgment precision in the case of judging the server which was processing at the end using the local state variable concerned.
[0027] This effectiveness to the above-mentioned server priority record step The following four steps, Namely, the 1st server priority record step which changes the server priority of a self-server so that the highest priority may be shown when the condition of a self-server changes in the single master condition, The 2nd server priority record step which changes the server priority of a self-server so that the 2nd priority may be shown when the condition of a self-server changes to master mode, The 3rd server priority record step which changes the server priority of a self-server so that the minimum priority may be shown when the condition of a self-server changes to a slave state, When the condition of a self-server changes to a idle state, it becomes much

more remarkable by giving the server (recording-again-former priority and equivalence) priority maintenance step which inhibits modification of the server priority of a self-server.
[0028] With the configuration which applied the four above-mentioned server conditions, a failure occurs in two sets of both servers. When both servers of both return from a failure after that, it is based on the judgment result and state transition diagram in the above-mentioned final-treatment computer judging step. It is good to perform the state transition to which one side of the two above-mentioned sets of servers will be in a single master condition from a idle state, and another side will be from a idle state in a slave state, and to perform the state transition from which above-mentioned one side will be in master mode, while the account another side of Gokami has been a slave state.
[0029] moreover, when a failure occurs in two sets of both servers and only a self-server returns from a failure after that It is based on the judgment result and state transition diagram in the above-mentioned final-treatment computer judging step. [whether the state transition to which a self-server will be in a single master condition from a idle state is performed, and] Or when a current condition is maintained and a partner server returns from a failure after that, it is good to perform the state transition to which the partner server concerned will be from a idle state in a slave state, or to perform the state transition from which one side will be in a single master condition, and another side will be in a slave state.
[0030] Moreover, since the server priority of the self-server concerned is not the highest priority when a failure generates this invention to the above-mentioned state-transition step at both following step (the two above-mentioned sets of i.e., servers) and only a self-server returns from a failure after that In the condition that cannot perform a judgment at the above-mentioned highest priority judging step, but the self-server concerned continues processing and that it cannot judge with it being a computer When the compulsive initiation instruction which makes processing continue compulsorily to the self-server concerned is given from the outside, the self-server concerned is characterized also by giving the compulsive initiation step to which the state transition of the self-server concerned is made to perform as a server which continues processing.
[0031] Since the server priority of one set is not the highest priority even if a failure occurs in two sets of servers and only one set carries out a failure return after that in such a configuration It is impossible to judge with a priority being higher than one more set which has the direction of one of them in a idle state. Therefore, even when one of them cannot judge with it being the server which was processing at the end, it becomes possible by giving a compulsive initiation instruction from an external computer (computer for command transmission) to one of them to make it process compulsorily. Thus, even when it is impossible, the function in which processing can be made to start compulsorily is convenient [the continuation of processing], when it is more convenient to resume processing.
[0032] Here, if it is made to process compulsorily to the server which carried out the failure return when the priority of the server in a idle state is the highest priority, both servers of both will serve as the highest priority. Although a server with a high priority cannot be specified, therefore processing can be automatically taken over in neither of such the condition when a failure occurs in both servers and a failure return is carried out after that [both], management becomes possible easily by performing compulsive initiation again.
[0033] In addition, this invention concerning the record medium which recorded the program which judges the server which was processing at the above-mentioned last is materialized also as invention concerning the sex computing system for Takayoshi equipped with a functional means by which each computer is realized by the program concerned, and is materialized also as invention concerning an approach with the procedure applied by the program concerned.
[0034]
[Embodiment of the Invention] Hereafter, with reference to a drawing, it explains per gestalt of operation of this invention. Drawing 1 is the block diagram showing the configuration of the sex computer system for Takayoshi concerning 1 operation gestalt of this invention.
[0035] In the system of drawing 1 , two Servers (server computer) 100a and 100b and the computers 300 for command transmission are connected to the network 500. Moreover, the client (client computer) which receives the service from Servers 100a or 100b and which is not illustrated is also connected to the network 500. The disk units (represented by the hard disk drive unit) 200a and 200b as external storage are connected to Servers 100a and 100b, respectively. This computing system does not have shared memory equipment.
[0036] Even if a failure occurs in the server which is processing by one side of the two sets of Servers 100a and

100b processing, the computer system of drawing 1 Another server not only can succeed processing, but a failure occurs in both the servers 100a and 100b of both. Also when at least one side returns from a failure after that, the description is to have the structure which can judge correctly the server which succeeds processing, i.e., the server which was processing at the last. About this structure, it mentions later.
[0037] In addition, the conditions including cutting of a power source etc. that "generating of a failure" in this operation gestalt becomes impossible as for the continuation of processing else [, such as a hardware failure and a software failure,] are said. Moreover, it says that "a return from a failure" will be in the condition that processing other than the return from a hardware failure and the return from a software failure can be continued, including the injection of a power source etc.
[0038] Now, in order to enable taking over of processing, it not only performs the only same processing by another server, but it must receive information required to continue processing from the server which was processing till then. However, in the system which does not have shared memory equipment like this operation gestalt, since information transfer between the servers through the shared memory equipment concerned cannot be performed, the server which performs processing must send information required for taking over to the server of another side. However, by this method, by the relation between the generating stage of a failure, and an informational transfer stage required for taking over, as the column of [Description of the Prior Art] described, it may become difficult to judge the server which was processing at the end. Then, although the "approach of limiting taking over at a time" stated by [Description of the Prior Art] or the "approach of using time information" is learned as an approach which the server which was processing at the end can judge easily, unattended operation cannot be realized or there is a problem that judgment precision (integrity) is inadequate.
[0039] Unattended operation is explained in full detail here. Unattended operation points out the operating method which continues processing as much as possible, whenever failure generating or a failure return may take place to the server of one of the two or both, as the column of [Description of the Prior Art] described. In order to realize unattended operation, it can be necessary to perform next actuation (1) - (5) at least.
[0040] (1) When a failure occurs in two sets of servers and two sets of servers return after that, judge whether processing is possible, when possible, determine the server which should process, and start processing. One more set of a server is changed into the condition which can be taken over. Even if a failure occurs in the server which is carrying out delivery and processing altogether to one more set of a server from the server which is processing information required in order to succeed processing, saying "it changes into the condition which can be taken over", it points out that one more set of a server enables it to continue processing.
[0041] (2) When a failure occurs in two sets of servers and one set of a server returns after that, judge whether processing by the server which returned is possible, and when possible, start processing.
[0042] (3) One set of a server is processing, and when it is in the condition which the failure has generated and the server which the failure has generated returns, change one more set of a server into the condition that the server can be taken over.
[0043] (4) When one set of a server is processing, it is in the condition that one more set of a server can be taken over and a failure occurs in the server under processing, take over.
[0044] In addition, the next actuation (condition) must be considered.
(5) A failure generates every timing.
[0045] Based on these actuation (condition), the state transition diagram 800 shown in drawing 2 is created (preparation). Here, the condition of a server is classified into Master (master mode) M, the single master (single master condition) SM, Slave (slave state) SL, and four conditions of Halt (idle state) X. That is, the conventional master M is classified into two conditions of the new master M and the single master SM according to this operation gestalt, and the description is in the point that the condition of telling the single master SM to Master M, Slave SL, and three conditions of Halt X was added, with it.
[0046] The condition of a server is first divided by whether the server is performing processing or it is not performing, and becomes like the :slave SL which omits the :master M which is processing, and single master SM processing, and Halt X.
[0047] Furthermore, the partner who succeeds the processing in which the partner who succeeds the processing which is performing processing for each of "it is processing", and "not processing", and which is : Performed exists, and which is performing -- master M does not exist. -- The information for the :taking over which omits single master SM processing has received. -- The information for slave SL taking over has not received. -- It

classifies like halt X.
[0048] Since it does not process but taking over is also impossible when a failure occurs in a server so that clearly, the condition of the server concerned will be in the condition of Halt X.
[0049] Now, in the state transition diagram 800 of drawing 2 , if the condition of another [A and] server (for example, server 100b) is set to B, (A B) will be written. [the condition of one server (for example, server 100a)] (A B) and (B A) express the condition of differing.
[0050] Each condition of being applied with the state transition diagram 800 of drawing 2 is explained. The condition that both servers are in a idle state first [both] (X X) is shown. Both of the servers and processing are not performed, either but, specifically, the condition that taking over is also impossible is shown.
[0051] (SM SL) and (SL SM) have one server in SM (single master) condition, and show the condition that the server of another side is in SL (slave) condition. In order to change into the condition which can be taken over, specifically, the condition of having sent the information for taking over processing to the server of the condition of SL is shown from the server of the condition of SM. In this condition, although processing is performed by one server, the taking over by the server of another side cannot be performed.
[0052] (SL M) and (M SL) have one server in M (master) condition, and show the condition that the server of another side is in SL (slave) condition. It is in the condition which processing is performed by one server and can specifically be succeeded by the server of another side.
[0053] (X SM) and (SM X) have one server in SM (single master) condition, and show the condition that the server of another side is in X (halt) condition. Although one set of a server is processing, specifically, it is in the condition that information for taking over to one more set of a server cannot be sent. That is, taking over is impossible.
[0054] Next, with reference to the state transition diagram 800 of drawing 2 , above-mentioned actuation (1) - (5) is explained.
(1) Of operation: The condition that the failure has occurred and stopped by both 2 set servers is in the condition of (X X). Here, if both servers return from a failure, the state transition of 1-1-1->1-1-2 of drawing 2 or 1-2-1->1-2-2 will be performed.
[0055] Thus, when both servers return from a failure, processing is first begun by one of servers. And in order to change into the condition which can be taken over, it changes in the condition of sending information required for taking over to one more set of a server from the server which is processing (1-1-1 or 1-2-1). this -- or (SL SM) (SM SL) it is a condition.
[0056] then, the condition of [if taking over becomes possible / one side / with a slave state] master mode (to a single master condition) in another side -- that is, (SL M) -- or (M SL) moves to a condition (1-1-2 or 1-2-2). Information required for taking over also in this condition is sent to a slave from a master.
[0057] Actuation (2): When only one server returns [a car server] from a failure in the state of failure generating (= halt) (X X), transition of 2-1 or 2-2 is performed. This is a state transition for changing the server which returned into the condition (condition of a single master) that processing is possible, or (X SM) (SM X) it moves from it to a condition.
[0058] : of operation (3) -- the condition that one server is processing and the server of another side calls it failure generating (= halt) -- or (X SM) (SM X) -- it is . this (X SM) -- or (SM X) the case where the server of a idle state returns from a failure in the condition -- respectively -- 3-1-1->3-1 -- the state transition of -2 or 3-2-1->3-2-2 is performed.
[0059] In this way [or (X SM) (SM X)], when the server of a idle state returns from a failure in the condition, first, with a single master condition, one side moves from another side to the condition of calling it a slave state (from a idle state) (3-1-1 or 3-2-1), and sends information required for taking over to another server (slave) from the server (single master) which is processing. this -- or (SL SM) (SMSL) it is a condition.
[0060] then, the condition of [if taking over becomes possible / like actuation (1) / one side / with a slave state] master mode (to a single master condition) in another side -- that is, (SL M) -- or (M SL) moves to a condition (3-1-2 or 3-2-2). This is a state transition for establishing a master and a slave by making the server which returned into a slave, making as a master the server which was processing till then.
[0061] : of operation (4) -- saying [the condition which can be taken over] -- or (SL M) (M SL) -- ** -- it is being in the condition of saying. the case where a failure occurs in the server of a slave state in this condition -- or (X SM) (SM X) -- ** -- it changes in the condition of saying (4-1 or 4-2). This is a state transition in the

condition of not taking over, although the server same as it is continues processing. on the other hand, a condition when a failure occurs in the server of master mode (SL M) (M SL) -- or (SM X) (X SM) changes in the condition (4-3 or 4-4). This is a state transition for the server which was a slave state till then to succeed processing.
[0062] Actuation (5): When a failure occurs in a server, the server will surely be in a idle state (X). This is 5-1 thru/or 5-8. One side will be in the condition of explaining each actuation below and that another side carries out transition of 5-1 or 5-5 when it gets or (M SL) blocked (SL M) and a failure occurs in both servers in the condition, the condition of a slave, and (X X), by the master first.
[0063] Next, or (SM SL) has sent the information for making taking over possible from the server of a single master condition to the server of a slave state (SL SM), when a failure occurs in the server of a single master condition in the condition, since taking over is impossible, transition of 5-2 or 5-6 is performed, and both of the servers will be in the condition of a idle state, and the condition of getting it blocked (X X). moreover, the above (SL SM) -- transition of 5-2 or 5-6 will be performed, and, as for both servers, or (SMSL) will be in a idle state, also when a failure occurs in both servers in the condition.
[0064] moreover, the above (SL SM) -- in the condition, transition of 5-3 or 5-7 is performed for or (X SM) (SM X), and or (SM SL) moves to a condition, when a failure occurs in the server of a slave state. In this condition, processing is continued by the server (server of a single master condition) which remained.
[0065] next, one server -- a single master condition -- another server -- a idle state -- that is, (X SM) -- or (SM X) -- the case where a failure occurs in the server of a single master condition in the condition -- transition of 5-4 or 5-8 -- carrying out -- both servers -- a idle state -- it becomes that is, (X X).
[0066] As mentioned above, actuation [for performing unattended operation] (1) - (5) is altogether contained in the state transition diagram 800 shown in drawing 2 .
[0067] Now, about above-mentioned of operation (1) actuation (3) - (5) among - (5), since it is decided which will be the server which changes a condition, controlling is easy.
[0068] However, the server to which it needed to judge whether the server to which actuation (1) takes over processing was self, and actuation (2) returned it from the failure also needs to judge whether you may succeed processing as it is. This operation gestalt has the description in the point of having devised the technique of judging the server which succeeds processing in this actuation (1) and (2) using the classification of the server condition described previously. The detail of this judgment technique is explained separately. Here, since processing is succeeded and a server turns into a server of the direction which was processing to the last, that is described.
[0069] First, in actuation (1), the conditions which are the server which should succeed processing are the server of the direction which was processing to the last between two sets of servers. The reason is explained below.
[0070] In this operation gestalt, in order for the server which was not processing by then to continue processing, it is necessary to receive the information for taking over from the server which was processing. When the information for taking over is not sent to another server from the server concerned just before a failure occurs in the server which was processing at the end, processing cannot be continued in another server. Therefore, it is necessary to make the server of the direction which was processing at the end into the server which should succeed processing.
[0071] Next, when the information for taking over is sent to another server from the server concerned before the failure occurred in the server which was processing at the end, it seems to either of the servers that it is possible to carry out a continuation of processing. However, about the processing just before failure generating, the failure may have occurred, before sending information required for taking over of the processing concerned to above-mentioned another server. Therefore, also in here, it is necessary to make it into the server which should succeed processing for the server of the direction which was processing at the end.
[0072] As mentioned above, in actuation (1), the server which was processing to the last needs to succeed processing.
[0073] Next, in actuation (2), the conditions which make the server which returned from the failure process are the server of the direction where the failure return server concerned was processing to the last two sets of inside for the same reason as the case of actuation (1).
[0074] Now, in each server, there is "an approach using time information" which stated as technique to judge

whether it is the server to which self was processing to the last in the column of [Description of the Prior Art]. However, "the approach using time information" assumed a global thing called time amount tacitly, and it is materialized under assumption that the clock which each server uses always synchronizes, and since the actual clock does not always not necessarily synchronize, a problem is in integrity.
[0075] For this reason, with this operation gestalt, the technique of judging the server which was processing at the end only using local information is applied. However, there are the following problems in local information. If a server with the information concerned stops, since the information with local it cannot be changed from other servers, it is the problem that the information which each server has may be contradictory.
[0076] About this problem, the case where the server which is processing has locally the information on a purport that it is processing by itself is stated to an example. In this example, the server which is processing has the information "he is processing [be / it]." Then, when a failure occurs in this server and a partner server succeeds processing, another server will also have the information "he is processing [be / it]." At this time, the information which the server which the failure generated has "he is processing [be / it]" cannot be rewritten. Suppose that the failure occurred also in another server and the server of both after that returned from the failure in such the condition this time. At this time, both servers will have the information "he is processing [be / it]." Therefore, it cannot judge which was processing at the end only by the server which is processing having locally the information "he is processing [be / it]."
[0077] Then, he controls based on the state transition diagram 800 having shown the condition of each server classified into four conditions including a single master condition according to this operation gestalt in drawing 2 R> 2, and is trying to solve this problem by having the state variable which can take three conditions called the server priority mentioned later as local information.
[0078] Here, if drawing 1 is referred to again, Servers 100a and 100b have the same configuration. That is, Servers 100a and 100b are all equipped with the cluster Management Department 110, processing / taking over section 400, and the server priority processing section 700. These each part 110,400,700 is functional means realized when Servers 100a and 100b read and perform a predetermined software program. It is recorded and provided for a record medium with here same the software (cluster software) for realizing the cluster Management Department 110, the software (processing software) for realizing processing / taking over section 400, and the server priority management software for realizing the server priority processing section 700, for example, CD-ROM, and is installed and used for the disk units 200a and 200b which Servers 100a and 100b have. In addition, each above-mentioned software may be beforehand installed in disk units 200a and 200b, and it may be recorded on the respectively separate record medium. Moreover, you may download through a network 500.
[0079] The cluster Management Department 110 has the function to perform state-transition control according to the state transition diagram 800 shown in drawing 2 . That is, the cluster Management Department 110 performs state-transition control of two sets of Servers 100a and 100b based on the state transition diagram 800 defined beforehand by communicating through another cluster (it operating on server) Management Department 110, and a network 500. Moreover, the cluster Management Department 110 sends the change-of-state information 901 which shows the newest transition state to processing / taking over section 400 and the server priority processing section 700 in the case of a state transition.
[0080] Moreover, the cluster Management Department 110 is in the condition which has both the servers of Servers 100a and 100b in a idle state (X), and when the server (self-server) to which self operates returns from a failure, it sends the change-of-state information 901 which shows the failure return of a self-server to the server priority processing section 700. Furthermore, by the communication link with the cluster Management Department 110 of a partner server, the cluster Management Department 110 detects failure generating of a partner server, and the return from a failure, and performs a state transition. In addition, at the time of the failure return of a self-server, the cluster Management Department 110 waits for the predetermined time after a return, investigates whether the partner server also returned from the failure by the communication link with the partner server concerned, and is made to send the change-of-state information 901 to after an appropriate time.
[0081] Processing / taking over section 400 controls initiation/halt of actuation for processing and processing taking over according to the change-of-state information 901 acquired from the cluster Management Department 110 (the newest transition state is shown). Processing / taking over section 400 consists of a processing control section 410 and a taking over control section 420.

[0082] The processing control section 410 controls initiation/halt of processing, and the taking over control section 420 controls initiation/halt of actuation for taking over of processing. Processing is activation of application programs, such as DBMS (database management system).
[0083] The server priority processing section 700 records the priority (server priority) 210 of self-server 100i on disk unit 200of self-server 100i (i is a or b) i (secured priority record section), and measures the priority 210 concerned with the priority 210 recorded on disk unit 200of partner server 100j (j is a or b, however i!=j) j. This server priority 210 is information required in order to judge the server which was processing to the last, and takes either of the tri-states of 1, 2, and 3.
[0084] The server priority processing section 700 consists of the condition write-in section 710 and the comparison processing section 720. The condition write-in section 710 updates the priority 210 of self-server 100i currently recorded on disk unit 200of self-server 100i i (secured priority record section) according to the change-of-state information 901 acquired from the cluster Management Department 110.
[0085] When the change-of-state information 901 which shows the failure return of a self-server is received from the cluster Management Department 110, a priority 210 is delivered [the comparison processing section 720] and received through a network 500 between the comparison processing sections 720 of a partner server. And the comparison processing section 720 is measuring the priority 210 of self-server 100i, and the priority 210 of partner server 100j, and judges whether the direction of self-server 100i has a high priority. The detail of this judgment is mentioned later. Priority comparison / judgment result in the comparison processing section 720 is used for the judgment (that is, is self-server 100i a server used as a master or not?) of whether to be the server which is performed at the cluster Management Department 110 and to which self-server 100i processed at the end.
[0086] The computer 300 for command transmission is transmitted to the cluster Management Department 110 of a server where the instruction concerned specifies the compulsive initiation instruction demanded by the user among Servers 100a and 100b through a network 500. This compulsive initiation instruction is an instruction which makes processing start compulsorily by the appointed server.
[0087] The technological background which introduces the compulsive initiation function of processing using a compulsive initiation instruction here is described. When a failure occurs in both the servers of Servers 100a and 100b and only one server 100i returns from a failure after that that is, in the above-mentioned actuation (2), it may be judged with the failure return server 100i not being the server which should process. It is possible to wait until the server which should process essentially [another] returns as actuation in this case.
[0088] However, since the twist which processing has stopped until another server returned does not perform a continuation of processing, also when it is more convenient to resume processing by server 100i which returned from the failure previously, naturally there may be.
[0089] So, with this operation gestalt, it can be made to carry out selection assignment of one of the modes out of two sorts of modes, the compulsive initiation mode which can start processing compulsorily by the server which carried out the failure return previously in such a case, and the standby mode for which it waits until the server which should process essentially returns from a failure. Here, only when it is usually set automatically by the standby mode and a compulsive initiation instruction is given from the computer 300 for command transmission, the configuration changed to compulsive initiation mode is applied.
[0090] Next, the detail of actuation of Servers 100a and 100b is explained focusing on actuation of the server priority processing section 700. Processing / taking over section 400 will perform the initiation or a halt of operation for the initiation of processing which describes the change-of-state information 901 concerned below according to the newest condition which reception and the change-of-state information 901 concerned show, a halt, or taking over, if the change-of-state information 901 which shows the newest transition state from the cluster Management Department 110 is sent.
[0091] First, about initiation of processing, and a halt, the processing control section 410 in processing / taking over section 400 operates as follows according to explanation of each condition of the master mentioned above, a single master, a slave, and a halt.
[0092] It processes, when self-server 100i changes into the condition of a master (M) or a single master (SM). On the other hand, when self-server 100i will be in a slave (SL) or a idle state (X), it does not process.
[0093] Next, about initiation of actuation for taking over, and a halt, the taking over control section 420 in processing / taking over section 400 operates as follows. One side performs transfer of the information for

processing taking over through a network 500, after a single master or a master, and another side have changed into the condition of a slave. Here, it becomes the informer of the information for taking over of a single master or the taking over control section 420 of a master side, and the taking over control section 420 of a slave side serves as a sink of the information concerned. Information transfer for taking over is not performed in the combination of other conditions.
[0094] Next, it divides into each actuation of the condition write-in section 710 which constitutes the server priority processing section 700, and the comparison processing section 720, and the detail of actuation of the server priority processing section 700 is explained in order. First, the condition write-in section 710 in the server priority processing section 700 performs server priority modification (record) processing in which the server priority 210 currently recorded on disk unit 200of self-server 100i i (secured priority record section) is changed, as follows according to the flow chart of drawing 3 according to the newest condition which the change-of-state information 901 concerned shows, when the change-of-state information 901 which shows the newest transition state from the cluster Management Department 110 is sent.
[0095] When the condition of self-server 100i changes in the single master condition (SM), it changes into 1 (steps S1 and S2). When the condition of self-server 100i changes to master mode (M), it changes into 2 (steps S1 and S3). When the condition of self-server 100i changes to a slave state (SL), it changes into 3 (step S1, S4). When the condition of self-server 100i changes to a idle state (X), it does not change (steps S1 and S5).
[0096] Here, the initial value of the server priority 210 differs by Servers 100a and 100b, and in one server, it is set as 2, and it is set as 3 by another server. Initial setting of this server priority 210 is performed by the processing before beginning unattended operation.
[0097] Next, when carrying out with the case where compulsive initiation is not performed, sequential explanation is divided and given about actuation of the comparison processing section 720 in the server priority processing section 700, i.e., priority comparison / judgment actuation required to judge the server which was processing at the last at the cluster Management Department 110.
[0098] First, the actuation when not performing compulsive initiation is explained with reference to the flow chart of drawing 4 . That it is necessary to judge the server which was processing at the end in the state transition diagram 800 shown in drawing 2 When a failure occurs [both] in the servers 100a and 100b of two, the case of 1-1-1->1-1-2 or transition of 1-2-1->1-2-2, and the case of 2-1 or transition of 2-2, i.e., two sets, and it has become a idle state (X X) They are the case where both servers return from a failure, and the case where one of servers returns from a failure. In addition, in the following explanation, the intersection is taken and 1-1-1->1-1-2 and 1-2-1->1-2-2 are expressed 1-1 and 1-2. The cluster Management Department 110 of Servers 100a and 100b will send the change-of-state information 901 which shows that to the server priority processing section 700 of a self-server, if a self-server carries out a failure return in the state of (X X).
[0099] The comparison processing section 720 in the server priority processing section 700 When the change-of-state information 901 which shows that the self-server carried out the failure return from the cluster Management Department 110 of a self-server is sent, for the above 1-1, transition of 1-2, 2-1, or transition of 2-2 It is judged as what the comparison and the judgment of the server priority 210 required for the judgment (judgment of whether a self-server succeeds processing) of whether the self-server was processing at the end were required as, and a comparison and the judgment concerned are performed as follows.
[0100] When becoming transition of 1-1 or 1-2: (1) comparison processing section 720 communicates through the comparison processing section 720 and the network 500 of a partner server (referred to as server 100j), delivers and receives the server priority 210, is with a self-server (referred to as server 100i), and partner server 100j, and compares and judges which server priority 210 is high (steps S11 and S12). As for the server priority 210, 1 is the highest, and it becomes low in order of 2 and 3 here. that is, the server priority 210 -- by 1, 2 shows the 2nd priority and 3 shows the minimum priority for the highest priority. In addition, it is also possible to judge with the direction of self-server 100i having a high priority, without acquiring the server priority 210 of partner server 100j, when the server priority 210 of self-server 100i is 1. About this reason, it mentions later.
[0101] (2) Notify the cluster Management Department 110 concerned of the comparison processing section 720 as a response to the change-of-state information 901 to which it was sent in the condition from the cluster Management Department 110 of a self-server as a result of [of the server priority 210] the comparison / judgment (i.e., a server priority judging result with a priority high [the direction of self-server 100i]) (X X) (step S13).

[0102] (3) The cluster Management Department 110 will do the state transition also of the judgment result concerned and the partner server 100j as follows according to the flow chart of drawing 5 based on whether the failure return is carried out and a state transition diagram 800, if a server priority judging result is received from the comparison processing section 720.
[0103] First, when it can judge with the direction of self-server 100i having the high server priority 210 by the server priority judging result from the comparison processing section 720 (steps S21 and S22), self-server 100i judges the cluster Management Department 110 to be the server which was processing at the end. And when partner server 100j is also carrying out the failure return (step S23), since self-server 100i finally becomes a master, the cluster Management Department 110 changes so that it may be in a single master condition (SM) first and partner server 100j may be in a slave state (SL) (step S24). On the other hand, not as the server which was processing at the end but as a thing from which partner server 100j finally becomes a master, when the direction of partner server 100j has the high server priority 210, (step S25) self-server 100i changes so that self-server 100i may be in a slave (SL) condition and partner master 100j may be in a single master condition (SM) (step S26). That is, the state transition of 1-1-1 (SL SM) of drawing 2 or 1-2-1 (SM SL) is performed. In addition, when partner server 100j has not carried out a failure return, it becomes the transition of (step S23), 2-1 (X SM), or 2-2 (SM X) mentioned later (step S27).
[0104] (4) The cluster Management Department 110 sends this state-transition result (SL SM), i.e., the newest transition state, or (SM SL) the shown change-of-state information 901 to processing / taking over section 400 and the server priority processing section 700. The activity of processing / taking over section 400 (the inner processing control section 410 and the inner taking over control section 420) in this case and the condition write-in section 710 in the server priority processing section 700 is clear from previous explanation, and it will change so that the server of the higher one of the server priority 210 may finally serve as a master (if generate and there is no failure in Servers 100i or 100j again). That is, it changes in the condition of 1-1-2 (SL M) of drawing 2 , or 1-2-2 (M SL).
[0105] When becoming transition of 2-1 or 2-2: (1) comparison processing section 720 communicates through the comparison processing section 720 and the network 500 of partner server 100j, delivers and receives the server priority 210, is with self-server 100i and partner server 100j, and compares which server priority 210 is high. However, when becoming transition of 2-1 or 2-2, since partner server 100j is still in a idle state (X), it cannot acquire the server priority 210 of partner server 100j. However, if the server priority 210 of self-server 100i is 1, it can be judged with mentioning later that the direction of self-server 100i has a high priority.
[0106] (2) Since the comparison processing section 720 has partner server 100j in a idle state, when the server priority 210 of the partner server 100j concerned cannot be acquired there, judge whether the server priority 210 of (step S11) self-server 100i is 1 (that is, the highest priority) (step S14). When the server priority 210 of self-server 100i is 1, the comparison processing section 720 notifies the cluster Management Department 110 of self-server 100i of the server priority judging result which shows that the direction of self-server 100i has the high server priority 210, without performing the comparison with the server priority 210 of partner server 100j (steps S15 and S13).
[0107] On the other hand, only with the server priority 210 of (step S14) self-server 100i, when the server priority 210 of self-server 100i is not 1, the cluster Management Department 110 of self-server 100i is notified of the priority judging result which shows a judgment failure (indeterminate) noting that it cannot judge whether the direction of a self-server has a high priority (steps S16 and S13). In addition, when the server priority 210 of self-server 100i is 3, self-server 100i can judge with the direction of partner server 100j having a high priority low [a priority] that is,. However, a state transition cannot be carried out when partner server 100j is in a idle state like this example. Then, when partner server 100j has not carried out a failure return, unless the server priority 210 of self-server 100i is 1, he is trying to treat as a judgment being altogether impossible with this operation gestalt.
[0108] (3) The cluster Management Department 110 will do the state transition also of the judgment result concerned and the partner server 100j as follows according to the flow chart of drawing 5 based on whether the failure return is carried out and a state transition diagram 800, if a server priority judging result is received from the comparison processing section 720.
[0109] First, the cluster Management Department 110 changes according to a state transition diagram 800, so that self-server 100i may become a single master, noting that it is in the condition in which partner server 100j

has not carried out a failure return and (steps S21-S23) self-server 100i is the server which was processing at the end, when the direction of self-server 100i has the high server priority 210 (step S27). On the other hand, a state transition is not performed until (step S25) partner server 100j carries out a failure return when a judgment result shows a judgment failure (when partner server 100j does not carry out a failure return and the server priority 210 of self-server 100i is not 1).
[0110] Supposing a server condition changes based on the above-mentioned judgment according to the state transition diagram 800 shown in drawing 2 , the server priority 210 will change like drawing 6 . In this drawing, when the condition of another [a and] server is set to B and a server priority is set to b for A and a server priority, (A B) was written and [a b] is written to the bottom of it. [the condition of one server] [condition] (A B) and (B A) express the condition of differing. [a b] and [b a] express the case where it differs.
[0111] The reason for the ability to judge below the server which was processing at the end correctly by the above-mentioned diagnosis is explained.
Proposition 1 The high server of the server priority 210 is a server which was processing at the end.
Proposition 2 If the server priority 210 of a certain server is 1, the server is a server which was surely processing at the end.
[0112] These two propositions 1 and 2 are proved. For that purpose, the following three lemmas 1-3 should just be realized.
[0113] 1 is set to the server priority 210 of self-server 100i when it is, lemma 1 assumption: and. Then, the server priority 210 of self-server 100i is not changed, and partner server 100j is not in master mode or a single master condition.
- The server priority 210 of partner server 100j is not 1.
Conclusion: When the server priority 210 of self-server 100i was 1, self-server 100i processed at the end.
[0114] 2 is set to the server priority 210 of self-server 100i when it is, lemma 2 assumption: and. Then, the server priority 210 of self-server 100i is not changed, and partner server 100j is not in master mode or a single master condition.
- The server priority 210 of partner server 100j is 3.
Conclusion: When the server priority 210 of self-server 100i was 2 and the server priority 210 of partner server 100j was 3, self-server 100i processed at the end.
[0115] Lemma 3 assumption: It is in the condition which the car servers 100i and 100j stopped.
Conclusion: The server priority 210 takes a mutually different value.
[0116] Certification of a proposition 1: First, use the conclusion of lemmas 1, 2, and 3 and prove a proposition 1. What is necessary is just to be able to prove that the server with a high priority was processing at the end in all the combination of the server priority 210, in order to prove a proposition 1.
[0117] When it is in the condition which both the servers 100i and 100j stopped from the lemma 3, since the server priorities 210 differ mutually, they are three kinds of the combination (2 (1 (1 2) 3) 3) of the server (it has tri-state of 1, 2, and 3) priority 210.
[0118] from a lemma 1 -- ******** (1 (1 2) 3) -- the server of 1 is processing [the server priority 210] at the end. Next, from the lemma 2, the server of 2 is processing [the server priority 210] (2, 3) at the end. Therefore, it can be said that the high server of the server priority 210 is processing at the end in all cases.
[0119] A proposition 2 is proved using certification: of a proposition 2, next the conclusion of lemmas 1 and 3. From the assumption of a proposition 2, the server priority 210 of self-server 100i is 1. At this time, the server priority of a partner server is not 1 from a lemma 3. At this time, it is the server to which self-server 100i was surely processing at the end from the lemma 1. Thus, if lemmas 1, 2, and 3 are realized, propositions 1 and 2 will be realized.
[0120] next, the lemmas 1, 2, and 3 -- it explains that will prove that a conclusion is realized if an assumption is realized, and the assumption of a lemma is finally realized about each.
[0121] Certification of a conclusion consisting of the assumption of a lemma 1: That 1 was set to the server priority 210 of self-server 100i means that self-server 100i was in the single master condition at the time. Then, I hear that partner server 100j is not processing once from the time, and partner server 100j has never been a master or a single master. Therefore, self-server 100i processed at the end.
[0122] The priority 210 of a self-server current in saying [that the priority 210 concerned is not changed after the time of 1 being set to the server priority 210 of self-server 100i] is 1. The server priority 210 of partner

server 100j current from an assumption is not 1. Therefore, the server priority 210 of self-server 100i is 1, and when the server priority 210 of partner server 100j was not 1, self-server 100i processed at the end.
[0123] Certification of a conclusion consisting of the assumption of a lemma 2: That 2 was set to the server priority 210 of self-server 100i means that self-server 100i was master mode at the time. In order to be in master mode, partner server 100j surely needs to be a slave state, therefore the server priority 210 of partner server 100j in the time is 3.
[0124] Then, I hear that partner server 100j is not processing once from the time, and partner server 100i has never been a master or a single master. Therefore, self-server 100i processed at the end.
[0125] Moreover, since the server priority 210 of partner server 100j does not change if it does not become a master or a single master, the server priority 210 of partner server 100j is still 3.
[0126] Moreover, the priority 210 of self-server 100i current in saying [that the priority 210 concerned is not changed after the time of 2 being set to the server priority 210 of self-server 100i] is 2.
[0127] Therefore, when the server priority 210 of self-server 100i was 2 and the server priority 210 of partner server 100j was 3, self-server 100i processed at the end.
[0128] What is necessary is just to prove taking the condition that the server priorities 210 always differ mutually, in order to prove that the server priority 210 takes the condition of always differing mutually when it is in the condition which the certification:car servers 100i and 100j of a conclusion being realized stopped from the assumption of a lemma 3. For that purpose, what is necessary is just to prove the following four assumptions 3-1 to 3-4.
[0129] Assumption 3-1: The initial value of the server priority 210 differs mutually.
When the server priority 210 of three to assumption 2 one of the two's server is 1, the server priority 210 of one of the two's server is not set to 1 any longer.
When the server priority 210 of three to assumption 3 one of the two's server is 2, the server priority 210 of one of the two's server is not set to 2 any longer.
When the server priority 210 of three to assumption 4 one of the two's server is 3, the server priority 210 of one of the two's server is not set to 3 any longer. If these can be proved, whatever the server priority 210 of one of the two's server may be, not becoming the same can already say the server priority 210 of one of the two's server.
[0130] The assumption 3-1 is clearer than actuation of the server priority 210. Below, assumption 3-2 to 3-4 is proved.
Certification of assumption 3-2: When the server priority 210 of one of the two's server (referred to as server 100p) is set to 1, server (referred to as server 100q) of one of the two is already a idle state. In order to set the server priority 210 of server 100q to 1 in this condition, without changing the server priority 210 of server 100p, server 100q must be changed into a single master condition. For that purpose, both servers must once change to a idle state, and server 100p must change further in a slave state and the condition that server 100q calls it a idle state. However, at this time, the server priority 210 of server 100p is 1, and this is contradictory in the decision approach of the state transition from a condition to the master slave condition which both servers stopped. Therefore, a partner server is unable to be in a single master condition, without changing the server priority 210 of a self-server. That is, when the server priority 210 of one of the two's server 100p is 1, the server priority 210 of one of the two's server 100q is not set to 1 any longer.
[0131] Certification of assumption 3-3: When the server priority 210 of one of the two's server 100p is set to 2, server 100q of one of the two is already set to 3 surely. Therefore, when the server priority of one of the two's server 100p is 2, the server priority of one of the two's server 100q is not set to 2 any longer.
[0132] Certification of assumption 3-4: When the server priority 210 of one of the two's server 100p is set to 3, server 100q of one of the two is already master mode. In order to set the server priority 210 of server 100q to 3 in this condition, without changing the server priority 210 of server 100p, server 100q must be made into a slave state. For that purpose, both the servers 100p and 100q must change in the stop condition, and server 100p must change further in a idle state and the condition that server 100q calls it a slave state. However, the server priorities 210 of both at this time are 3 and 2, and are [the direction of server 100q] high. [of a priority 210] This is contradictory in the decision approach of the state transition from a condition to the master slave condition which both servers 100,100j stopped. Therefore, a partner server is unable to be in a slave state, without changing the server priority 210 of a self-server. That is, when the server priority 210 of one of the

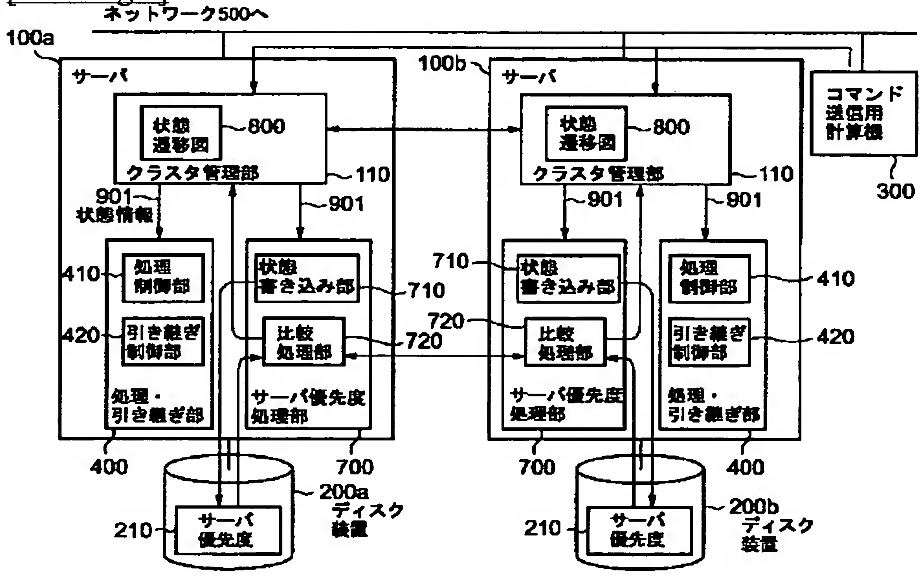
two's server 100p is 3, the server priority 210 of one of the two's server 100q is not set to 3 any longer. Thus, it has proved from the assumption that a conclusion was realized about lemmas 1, 2, and 3.
[0133] Next, it explains that the assumption of lemmas 1 and 2 is realized.
Certification of the assumption of a lemma 1 being realized: If the server priority 210 of partner server 100j is not 1 when 1 is set to the server priority 210 of self-server 100i, with reference to the state transition diagram 800 of drawing 2 , it will explain that partner server 100j cannot be in master mode or a single master condition, without changing the server priority 210 of self-server 100i.
[0134] After the server priority 210 of self-server 100i is set to 1, in order for partner server 100j to be in master mode, self-server 100i will need to be in a slave state, and the server priority 210 of self-server 100i will be then changed into 3. That is, partner server 100j is unable to be in master mode, without changing the server priority 210 of self-server 100i. Therefore, after the server priority 210 of self-server 100i is set to 1, partner server 100j is not master mode.
[0135] In order to set the server priority 210 of self-server 100i to 1 and for partner server 100j to become a single master after reliance From the state transition diagram 800 of drawing 2 , partner server 100j once A master, or [that self-server 100i will be in a slave or its condition that slave and self-server 100i calls / partner server 100j / it a master conversely] -- or Both the servers 100i and 100j must once be in a idle state, and partner server 100j must become a single master after that.
[0136] However, one side is a master, and when another side becomes a slave, since the server priority 210 of self-server 100i changes, it is impossible. Moreover, in order for both servers to be in a idle state and for partner server 100j to become a single master, the server priority 210 of partner server 100from the definition of judgment approach j must be 1. This is contrary to an assumption. Therefore, after 1 is set to the server priority 210 of self-server 100i, it can be said that partner server 100j is not in a single master condition. Partner server 100j cannot be in master mode or a single master condition from the above thing, without changing the server priority 210 of self-server 100i.
[0137] Certification of the assumption of a lemma 2 being realized: When the server priority 210 of self-server 100i is set to 2, explain that partner server 100j cannot be in master mode or a single master condition with reference to the state transition diagram 800 of drawing 2 , without changing the server priority 210 of self-server 100i.
[0138] The condition of self-server 100i in the time of the server priority 210 of self-server 100i being set to 2 is surely master mode, and the condition of partner server 100j is surely a slave state.
[0139] After the server priority 210 of self-server 100i is set to 2, in order for partner server 100j to be in master mode, self-server 100i will need to be in a slave state, and the server priority 210 of self-server 100i will be then changed into 3. That is, partner server 100j is unable to be in master mode, without changing the server priority 210 of self-server 100i. Therefore, after the server priority 210 of self-server 100i is set to 2, partner server 100j is not master mode.
[0140] After 2 is set to the server priority 210 of self-server 100i, partner server 100j is able to be in a single master condition, without changing the server priority 210 of self-server 100i. However, in order to set the server priority 210 of partner server 100j to 3, without changing the server priority 210 of self-server 100i after that, from the state transition diagram 800 of drawing 2 , both the servers 100i and 100j must change in the stop condition, and self-server 100i must change further in a idle state and the condition that partner server 100j calls it a slave state. However, as for the server priority 210 of both in this time, self-server 100i is contradictory in the decision approach of the state transition from a condition to the master slave condition that partner server 100j is 1 in 2, and both the servers 100i and 100j stopped this. Therefore, partner server 100j is unable to be in a single master condition, without changing the server priority 210 of self-server 100i. That is, after the server priority 210 of self-server 100i is set to 2, partner server 100j is not in the single master condition.
[0141] The server priority 210 of self-server 100i is set to 2 by the above, and partner server 100j has not become a master or a single master after reliance by it. Therefore, when it is in the condition which both the servers 100i and 100j stopped, the server priority 210 of one of servers can say that the server of 2, i.e., a priority, is a server to which the server with the higher server priority 210 was processing at the end when another server priority 210 was 3 in 2.
[0142] Next, the actuation in the case of performing compulsive initiation is explained. A state transition in case compulsive initiation is shown in drawing 7 , and transition of the server priority 210 are shown. The point

which is different from the case where there is no compulsive initiation which transition in case there is compulsive initiation shown in drawing 7 showed to drawing 6 is transition (transition shown by the arrow head of a thick wire in drawing 7) of 2-1 from the condition of (X X), or 2-2, and I hear that it has the transition which makes a single master compulsorily the server of the lower one of the server priority 210, and there is.
[0143] Therefore, it may have the server priority 210 [11] at the time of the tri-state of (X X), (X SM), and (SM X) (priority to which * mark is given in drawing).
[0144] For example, a server priority can perform a state transition by compulsive initiation to (X SM) in the state of (X X) at the time of [1 3] and [1 2]. A server priority is then set to [1 1]. When a failure occurs in the server of a single master condition in this condition, it is set to [1 1] by (X X).
[0145] When the condition that the server priority 210 of both servers is equal occurs as a result of compulsive initiation, it becomes impossible thus, to judge a server with a high priority. When both servers return from a failure, it becomes impossible therefore, to judge the server which was processing at the end. In this case, it is necessary to perform compulsive initiation once again.
[0146] Since there is the condition of calling it like the above [1 1] at the time of (X X) as a result of compulsive initiation, even if one server returns from a failure, it becomes impossible moreover, to perform automatically transition of 2-1 or 2-2. In order to carry out transition of 2-1 or 2-2, it is necessary to surely perform compulsive initiation once again.
[0147] Although processing is performed with the "master" in which the partner who succeeds the processing which processes and is performing the condition of a server in this operation gestalt exists as stated above The "single master" in which the partner who succeeds the processing which requires a line does not exist, The "slave" which has received the information for taking over although not processed, While performing the state transition of Servers 100a and 100b according to the state transition diagram 800 classified and created by four conditions of a "halt" of having not processed and having not received the information for taking over A state variable called the server priority 210 decided by transition of this condition is recorded on the local disk units 200a and 200b. Since the server which was processing to the last was judged using a state transition and a state variable (server priority 210) concerned, the following effectiveness can be acquired. First, when not using compulsive initiation, it can judge [one of the two of server 100a.100b or] whether both (that is, one [at least] server), no matter failure generating or a failure return may take place to what timing, when a failure occurred and returns to Servers 100a and 100b, it is the server to which the server concerned itself should process.
[0148] Actuation [for performing the above mentioned unattended operation especially in this operation gestalt] (1) - (5) It realizes by employing the cluster Management Department 110 which can do a state transition at two sets of within server 100a and 100b. The condition write-in section 710 in the server priority processing section 700 records the server priority 210 corresponding to a transition state for the judgment of a server required of actuation (1) and (2) at every state transition at a local disk unit. Unattended operation can be realized without using shared disk equipment and time information, since it has realized by measuring the server priority 210 of both the servers 100a and 100b in the comparison processing section 720 in the server priority processing section 700.
[0149] Moreover, even when it cannot judge with it being the server to which the server which returned takes over processing in actuation (2) when using compulsive initiation, processing can be succeeded compulsorily if needed. Also in this case, except for the case where both servers change in the special condition of calling it [1 1] by (X X), processing is automatically continuable. moreover, the above -- also in a special condition, processing is continuable using compulsive initiation again.
[0150]
[Effect of the Invention] As explained in full detail above, a state variable called the server priority which is decided by transition in the condition of following a predetermined state transition diagram according to this invention is recorded on a local store, and judgment precision can be improved, while losing a limit of the count of processing taking over and realizing unattended operation, since the server which was processing to the last was judged using a state transition and a state variable concerned.

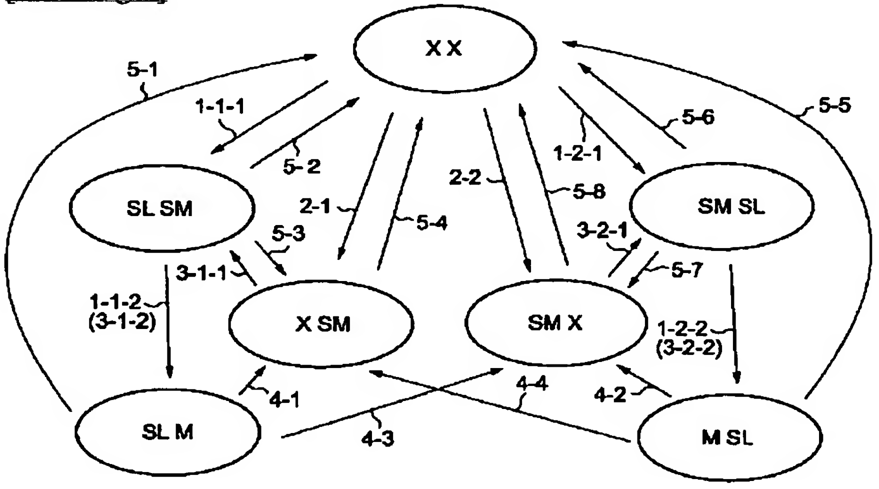
[Translation done.]

DRAWINGS

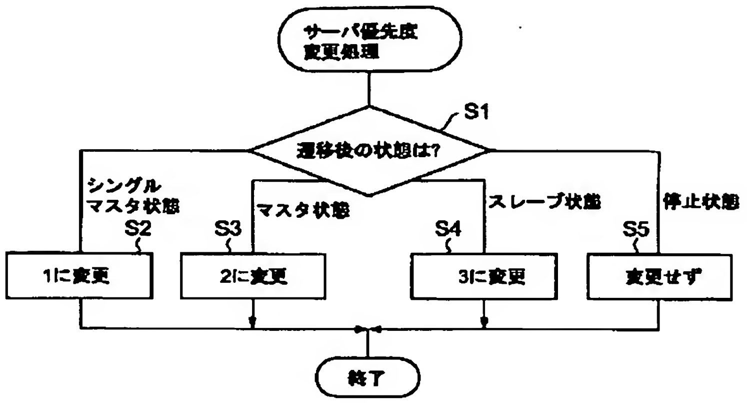
[Drawing 1]



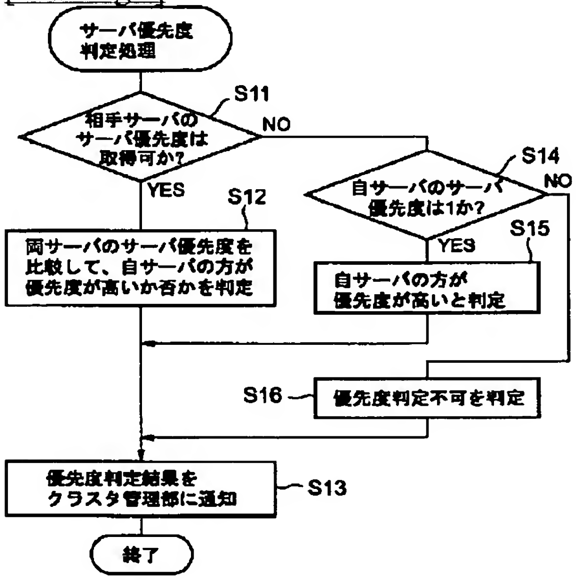
[Drawing 2]



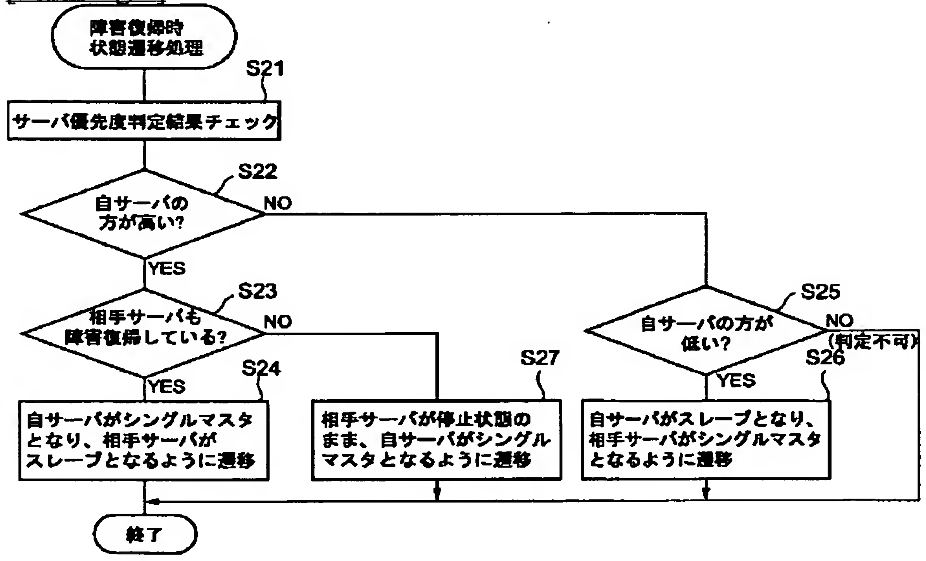
[Drawing 3]



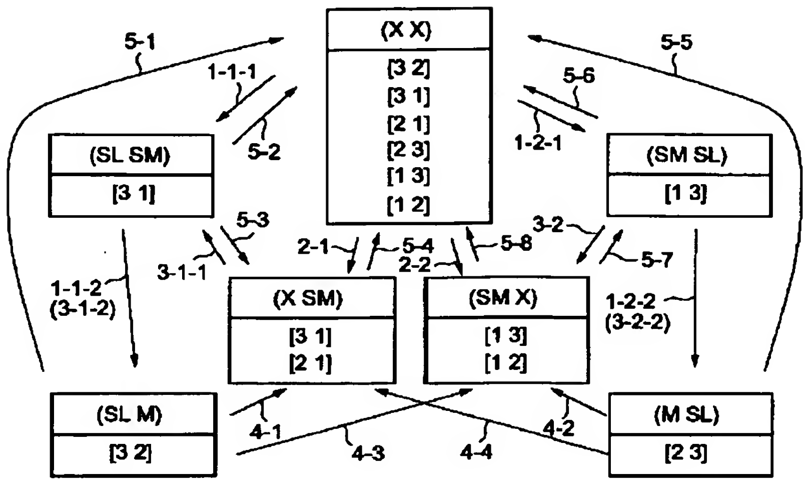
[Drawing 4]



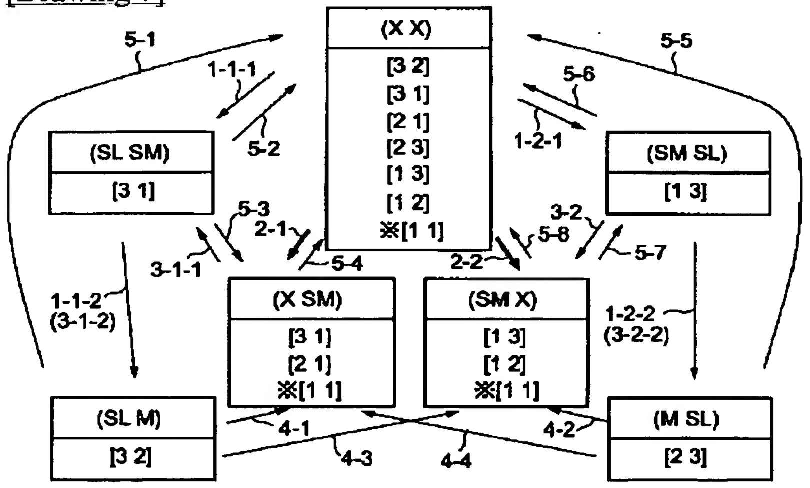
[Drawing 5]



[Drawing 6]



[Drawing 7]



[Translation done.]